# Cooperative Coevolution Applied to Dual-arm Robot Motion Planning

**Petar Ćurković, Bojan Jerbić,**
**Tomislav Stipančić**

*Faculty of Mechanical Engineering and Naval Architecture,University of Zagreb,*
*CROATIA (Tel: +385-1-6168422;).*
*e-mail: petar.curkovic@fsb.hr ; bojan.jerbic@fsb.hr ; tomislav.stipancic@fsb.hr*

**Abstract:** This paper presents a cooperative coevolutionary approach to path planning for two robotic arms sharing common workspace. Each arm is considered an agent, required to find transition strategy from given initial to final configuration in the work space. Since the robots share workspace, they present dynamic obstacle to each other. To solve the problem of path planning in optimized fashion, we formulated it to multi-objective optimization domain and implemented co-evolutionary algorithm to simultaneously optimize four conflicting objectives. End-effector trajectory length, end-effector velocity distribution, total rotate angle and number of collisions are the objectives to be optimized. Simulation results for two 2-R type robots are presented. Experimental verification is conducted on two FANUC Lr Mate 200iC robots.

*Keywords:* Planning, coevolution, multi-objective optimization, dual arm robot.

## 1. INTRODUCTION

Over the last two decades, evolutionary algorithms have been applied in a variety of fields, namely: robotics, scheduling, construction engineering, speech recognition, space engineering, image processing etc. An augmentation of evolutionary computation, coevolutionary computation, drives the inspiration from natural processes of coevolution between different (animal) species. The main difference between coevolutionary and standard evolutionary algorithms is in the nature of individuals' fitness evaluation. While the former uses a static fitness function for evaluation of individuals from a single population, the latter employs non-stationary fitness function for evaluating individuals from multiple populations, based on their interactions with individuals from other populations [1].

Coordinated path planning for multiple robots is difficult because the problem is NP-complete, whose complexity grows exponentially as the number of DOF increases [2] [3]. Required is to plan not only for the paths of individual robots, but also for the order of their consecutive movements, in order to prevent the robots from colliding with one another. The path planning problem for multiple manipulators sharing common work space have been studied for some time now [4].

Many important contributions to the problem of path planning in recent years have been made, each one possessing its own merits and disadvantages. Comprehensive survey can be found in [5]. Practical multi-robot motion planning problems are often decoupled in the sense that the robots trajectories are planned for only one robot at a time in priority order. Then in the second phase, velocities are modulated so that collisions between the robots are avoided. If a problem of path planning is represented as an optimization problem, robust optimization techniques, such as evolutionary algorithms have proven suitable for finding solution for such formulated problems. Rana and Zalzala [6] [7] propose an evolutionary planner to evolve near time-optimal collision-free trajectories for multi-arm robot manipulators. In this study, planning is carried out in joint space of the manipulator, and the path is represented as a string of via points connected by cubic polynomial splines. Davidor [8] also applies evolutionary algorithms to the trajectory generation by searching the inverse kinematics solutions to pre-defined end effector robot paths. Pires, Machado and Oliveira [9] employ multi objective genetic algorithm to evolve joint-space strings of manipulator configurations. Five indices, namely manipulator joints travelling distance, joint velocity, Cartesian distance, cartesian velocity and energy are used to qualify the evolving trajectories. Toyoda and Yano [10] used multi-purpose genetic algorithm to optimize movement of multi-joint robotic arms in presence of stationary obstacles. Optimum solutions with smooth trajectories and minimal joint rotation were obtained. Venegas and Marcial-Romero [11] present preliminary results of Constructive Solid Geometry based approach to path planning of multiple robot arms. They used two phase genetic algorithm to obtain a plan for the robotic arms by using a strategy that combines the exploration of the free collision space while looking for the target position from each previously explored area.

Majority of papers dealing with evolutionary – based path planning consider either single agent operating in an

environment without presence of obstacles, or in an environment containing static, point obstacles. The problem then boils down to finding suitable set of interior points, to be interpolated to formulate the polynomial of given order representing the trajectory.

The method presented in this paper considers concurrent development of robot trajectories for two 2R type robots sharing common work space. Each robot is considered an agent that is to find appropriate strategy for moving from given initial to final configuration. Since one agent presents dynamical obstacle to the other, and vice versa, often it is necessary for the agents to detour away from optimal, shortest paths, to find feasible strategies. It is also necessary to check for collisions between the links of the two robots in each consecutive time step.

Cooperative coevolutionary algorithms (CCEAs) offer great potential for concurrent multiagent domains. Two populations, each representing set of configurations of one robot in joint space, co-evolve to minimize number of collisions between interacting individuals, at the same time minimizing distance travelled, total joint rotation angle, and equalizing end-effector velocity profile. Best collaborators are sought and preserved to achieve memory effect in subsequent populations, and to bias coevolutionary search for optimal strategies.. The paper is organized as follows: Section 1 presents related literature and recent work, with focus on implementation of evolutionary algorithms to path planning. Problem and proposed algorithm are discussed in the section 2. Section 3 presents simulation and result for one given scenario. In section 4 some coevolutionary modifications are described.In sections 5. and 6 we discuss results and present initial experimental validation of presented approach. Section 7 concludes our work and gives motivation for future research.

## 2. PROBLEM AND ALGORITHM FORMULATION

In this paper, two *2-R* type robots with two links and two joints are considered. The end-effector is considered to move in the horizontal plane. The configuration spaces of the two robots are: $C_1 = q_{11} \times q_{21} \in R^2$ for the first robot and $C_2 = q_{12} \times q_{22} \in R^2$ for the second. The two manipulators are to move from given initial to a given final configuration. The lengths of all links are set to 1 *m*, with distance between the robots of 2.1 *m*. The links are free to rotate in the range $[0, \pi]\, rad$ .

### *2.1 Individual Representation*

An individual in each population is encoded as a string of real-valued vector in the joint space:

$$\left[\left\{q_{11}^{(\Delta t,G)},..,q_{ij}^{(\Delta t,G)}\right\},...,\left\{q_{11}^{((n-2)\Delta t,G)},..,q_{ij}^{((n-2)\Delta t,G)}\right\}\right] \qquad 1$$

Where *i* denotes the robot *i*=1,2, *j* is the number of DOF,$\Delta t$ is sampling time between two consecutive configurations, $q$ is angle between the link and positive x axis, G is current generation. At the beginning of the evolutionary process, joint values are randomly initialized, whereby the initial and final configurations are not encoded into the string because they remain constant throughout the search process. Without the lost of generality, adopted is normalized sampling time with $\Delta t$=0.1*s*.

### *2.2 Coevolutionary Algorithm Operators*

The algorithm starts with random initialization of two populations, each representing set of configurations for one robot. The performance of each robot's configuration depends on the current state of the robots from other population, since the two must coordinate the motion, to achieve continuous collision free movement. In the canonical CCEA, each individual from the first population should be evaluated by all individuals from other population, what is extremely time-consuming. To speed-up the evolution process, modified co-evolution is considered. In the modified version, each individual is evaluated by a finite set of the top collaborators from the other population, based on scores from previous generation. In this study, we evaluated the top 10 % of the populations, which is a modification of the approach proposed by the Sims [12], where "…the most "interesting" results occurred when the all vs. best competition pattern was used". The sizes of both populations were same and set to 60 individuals. In what concerns the selection operator, the successive generations are reproduced on the basis of roulette wheel selection. Standard single point crossover operator is used. The mutation operator replaces one allele with a given probability using the equation, where rand gives random number from the given interval:

$$q_{ij}^{(\Delta t,G+1)} = q_{ij}^{(\Delta t,G)} + rand \in \left(0, \pi/5\right] \Big| q_{ij}^{(\Delta t,G+1)} < \pi \qquad (2)$$

Defining adequate parameters of the evolutionary algorithms is known to be a difficult problem [13]. One problem is to develop an evolutionary algorithm and another problem is to make the algorithm to perform well on a specific problem. The connections between different operators are non linear and difficult to characterize. We have preformed a series of simulation experiments to experimentally determine a possible set of parameters performing well on the problem. There might be other combinations of parameters performing even better on the described problem.

### *2.3 Fitness Criteria*

Fitness criteria should take into account number of collisions between the two robotic arms, trajectory length, velocity profile, and total rotate angle. The most important criterion is the collision number, since collisions should be avoided at all costs. To check for collisions, in each time step $\Delta t$, linear system is solved that describes current positions of all links of the two robots.

### *2.4 Collision Penalty*

Collision penalty depending on collision between Robot 1 and Robot 2 in corresponding configurations is given by:

$$C_1 = \sum_{k=1}^{k=n-2} C_k, C_1 \to \min \quad (3)$$

where:

$$C_k = \begin{cases} 1 \text{ if R1 and R2 collide in i}^{th} \text{ generation} \\ 0 \text{ otherwise.} \end{cases} \quad (4)$$

It is important to note that the evaluation of the eq. 3 needs representatives from other co-evolving population and here is where cooperative coevolution makes contribution. At the same time, evaluation of potential collisions is the most time-demanding process in the algorithm.

*2.5 Total Distance of the End-effector Movement*

The distance travelled by the end effector should be minimized:

$$C_2 = \sum_{k=1}^{k=n} dist\left(p_j, p_{j-1}\right), C_2 \to \min \quad (5)$$

Where $p_j$ is the robot's intermediate end-effector position. In the case where no obstacles are present in the environment, optimal value of the function is length of the straight line connecting initial and final end-effector position. For all other cases, $C_2 > C_{2optimal}$.

*2.6 Total Rotation Angle*

Since the robots are redundant systems even in this simple form of 2 *DOF,* resulting in possibilities of reaching the same point in the space in elbow-up and elbow-down configurations, criterion of minimizing the total distance is not enough. Additionally, it is necessary to minimize the total rotation angle, to ensure no oscillations between the elbow-up and elbow-down configurations occur during the transition from initial to final configuration in the work space. Following expression defines total angle for one joint of one robot. Each robots' total angle should be minimized:

$$C_3 = \sum_{i=1}^{n} \left|\alpha_i - \alpha_{i-1}\right| \to \min \quad (6)$$

Where $\alpha_i$ is the angle between the limb of the robot and positive horizontal axis.

*2.7 End-effector velocity distribution*

To ensure even distribution of passing points along the robots' trajectory, the distance between two adjoining points in unit time should be equal:

$$C_4 = \left\{ dist\left(p_j, p_{j-1}\right)_{\max} - dist\left(p_j, p_{j-1}\right)_{\min} \right\} \to \min \quad (7)$$

Optimal value for $C_4$ is equal to 0, what means that all passing points are equally distant from each other, ande velocity is equally distributed along the trajectory.

*2.8. Objective Function Evaluation*

Objective (fitness) function for each candidate is calculated as weighted linear combination of above equations:

$$F = f\left(w_1 \cdot C_1 + w_2 \cdot C_2 + w_3 \cdot C_3 + w_4 \cdot C_4\right) \to \min \quad (8)$$

Where values of weight factors $w_i$ are constants. These values of weight constants have significant impact on the overall behavior of the algorithm. Namely, since objective criteria are in conflict, i.e. shortest distance criterion conflicts collision penalty criterion, proper tuning of these parameters is very tedious and time demanding. We are considering implementing non-linear functional relationships in weight parameters in the future. To address this problem in this paper, we implement dynamic weight factors, based on current distribution of individuals from each population and show the benefits of proposed approach.

## 3. SIMULATION RESULTS

Several experiments were conducted to test the performance of the proposed algorithm. It was determined that, beside the parameters of the fitness function, the success ratio of the algorithm depends on the initial and final configurations of the robots. The easiest scenario is when configurations of the robots result with no collisions. Most difficult scenarios occurred for configurations when significant detouring from shortest paths was necessary; (to ensure collision free motion). If there are no possible collisions between the robots on their consecutive stages, then the problem boils down to the problem of optimizing for only three criteria. If there are states where collisions can occur, it is four conflicting objectives to be simultaneously explored and satisfied, which increases the complexity of the problem significantly.

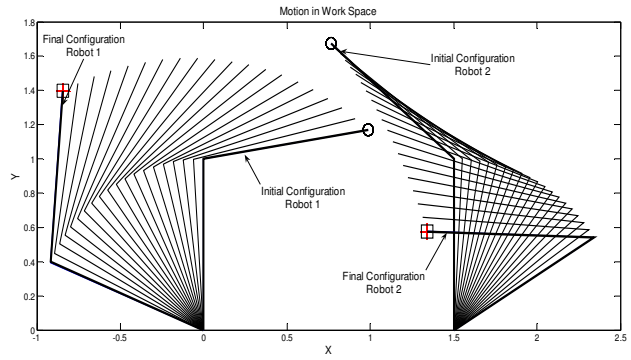


Fig. 1. Trajectories evolved for two robots simultaneously.

Fig. 1 shows evolved motions for two robots sharing work space. In the above case, all lengths of robot links are 1 *m*, and the distance between bases of the robots is set to 1.5 *m*. The algorithm successfully evolved trajectories for given start and end configurations of the robots. Trajectory of the left robot could be further optimized in terms of length, since it is obvious that end-effector performs arc motion, while linear motion would result in reduced trajectory length.

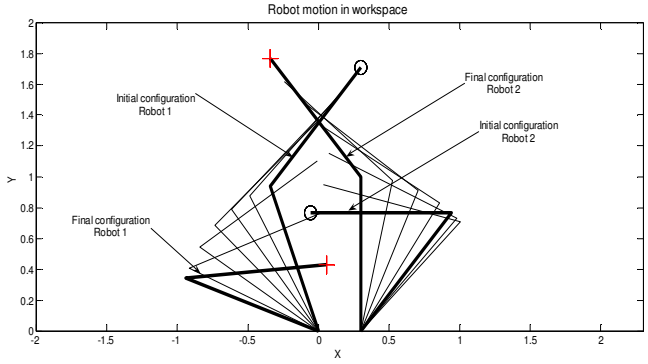


Fig. 2. Trajectories for the two robots for difficult initial conditions

Fig. 2 presents evolved motion for the two robots with different initial and final conditions. In this case, the configurations are chosen so to result in collision if the simple linear end effector motion would have been applied. This is the most difficult case for optimization, since, it is obvious that different optimization criteria are in conflict there is no unique global optimal solution. The two state spaces of the two robots are interconnected. Namely, problem of two robots sharing workspace differs from similar problems where dynamic or stationary obstacle is present in the work space. The fact that the two obstacles is actually controllable enables the simultaneous exploration and mutual adaptation of the two spaces by means of coevolutionary algorithm.

Operation of the coevolutionary algorithms is very complex and theoretical background is still developing in the research community [14] [15]. For example, it is not possible to employ simple elitist function to the coevolutionary algorithm, as it is in the case of standard evolutionary algorithm. The reason is that each individual from one population is, in canonical case, evaluated by each individual from coevolving population. The consequence is that performance of the individuals from first population depends on the structure of the other population(s). That means that individual, that had high fitness value and was good in one generation, may suddenly become not so good in the next generation. The fitness vs. time function is not monotonous in that case.

## 4. MODIFIED COEVOLUTION

Standard evaluation each individual by each individual from other populations is time consuming. To speed up the evolutionary process, we save top 10% individuals from both populations and evaluate them by each individual from other population. By doing so, we hope to only increase the speed of the convergence, without hindering the ability of the algorithm for finding solutions near Pareto front. Other issue is biasing coevolutionary search towards optimal solutions. Taking in consideration the nature of the problem, we introduce dynamic fitness function. Since it is very difficult to find proper combination of weight factors, we employ following procedure:

The search starts with weight parameters having initial values chosen either by random or by some previous experience. Afterwards, number of collisions is monitored for the pair of individuals we call *best collaborators* – the pair receiving the highest fitness value. Since the most important criterion is to find *collision free trajectories*, weight $w_1$ is increased and simultaneously all other weight values start to decay. This way, importance is given to part of the fitness function responsible for finding collision free paths. After such best collaborators are evolved, that have number of collisions equal to zero, the opposite process starts and other three weight factors start to increase, whereas $w_1$ starts to decay, as shown by Fig 3.

If, in any moment, best collaborators start to collide, the first process begins, increasing the importance of avoiding collisions.

This procedure enables guiding of the evolutionary process in desired direction very early. By evolving good individuals early, they can be improved in the later phase of the evolutionary process. Also, they are expected to seed good traits among the coevolving populations and increase the chance of finding global optima and to speed up the evolutionary process that is generally time-demanding. This is a kind of incremental learning approach, where the individuals are required to learn to walk before they learn to run.

If stationary fitness function is used, it is very difficult to evolve satisfactory strategies for the robots. There are numerous possibilities to improve this process of tuning fitness function. For example, we monitor only best collaborators i.e. one individual from the first and one individual from the second population to determine when to

change fitness function. At the other hand, gradients of the growth of weight factors are chosen after several experiments were conducted. It is very important parameter, whose behavior should be determined very carefully.

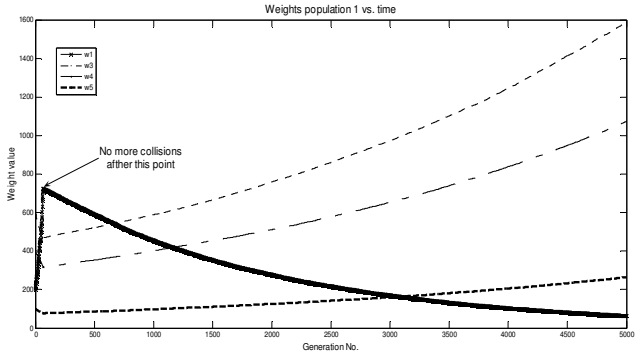


Fig. 3. Dynamic weight factors vs. generations.

## 5. DISCUSSION

As it is already mentioned, there are possibilities and ideas to further improve presented approach. There are two critical factors: speed of the convergence and completeness of the solution i.e. number of satisfying configurations developed by the algorithm before stopping criterion is matched. In what concerns speed of the convergence, it is proposed to evaluate a finite representative set from each population. This way, number of evaluations of the algorithm is decreased, but some, possibly good combinations of individuals might get lost. That is the reason why we keep the best collaborators, namely individuals from each population that, as result of their combination, maximize the composed fitness function.

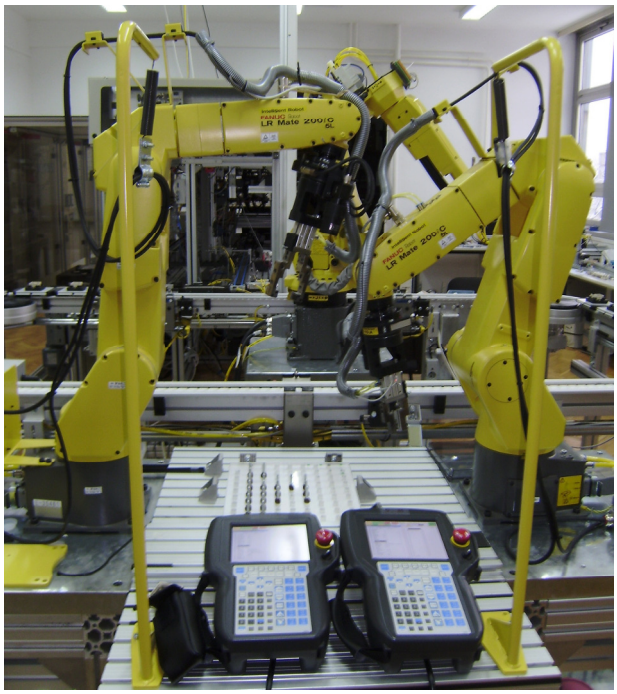

Fig. 4. Robots used for initial experiments

In what concerns completeness of solution, we propose dynamic weight factors for the fitness function. In such way, we are able to adapt the evolutionary process according to current state in the population. It is rather rough idea, how the process should look like, since only two individuals are monitored, and based on their interaction it is decided on the values of the weight factors. Simple method for changing fitness factors is adopted, namely, they are increased or decayed by adding or subtracting some fixed increment. The values of the increments remain fixed over the evolutionary process, but are different for each weight factor.

## 6. INITIAL EXPERIMENTAL VALIDATION

Initial experiments were conducted on the Laboratory equipment consisting of two six DOF Fanuc Lr Mate 200iC Robots. The robots are equipped with force sensors, cameras, and additional software packages enabling us to test various scenarios. Important is to notice that this two robots are controlled by separate controllers. Each robot with its own, what means they are not a priori synchronized in time domain. To synchronize the movements of the two robots a plug-in from FANUC named Robot Link is used. One robot is defined as master, while the other one is defined as slave. Master robot's controller is connected to the other robot via hub and constantly sends information regarding its movement to the other robot using fast Ethernet communication. Slave robot send acknowledge code (ACK) to the master, upon the motion is continued. If the communication between the two robots is lost in any moment during the motion, the robots will immediately stop.

Besides the Ethernet communication, Robot Link uses digital input and output (I/O) lines. These lines are used to confirm the end of synchronized motion and start of asynchronous motion.

For the experiment considered in this paper, only two degrees of freedom are allowed to change over the time, namely *J*1 and *J2*. The points calculated by coevolutionary procedure are fine-tuned and provided through teach pendant (see Fig. 3) to the robots. We intend to automate this procedure in the future. Upon defining the points and calibrating them to fit to the real environment, we defined intermediate points as (CNT XX) points, whereby XX defines how accurate the interpolation will be. The points that define start and end configuration of the robots are defined as FINE, what means they will actually be reached by the robots end-effector.

Since these are only initial experiments, and the research presented is in an early phase a lot of human intervention was needed to enable the robot to actually work.

## 7. CONCLUSIONS

Our long-term goal is the development of a framework based on described principles and its implementation to a pair of industrial robots, probably SCARA configuration. To fully automate the process of path planning, the algorithm should be further optimized in terms of speed in the first line. Off line path planning is also a possibility. In this case, initial and

final configurations should be memorized and connected to a database consisting of solutions for these configurations. If, at some point in the future, path planning is needed for known initial conditions, the solution should be retrieved from the database. There is a possible problem regarding the size of the database, and some discretization of the work space would be needed.

In this work, no physical characteristics of the robots (i.e. speed, acceleration etc.) are included in the simulation environment.

## ACKNOWLEDGEMENTS

This work is a part of the research project of the Croatian Ministry of Science, Education and Sports, *Autonomous Multi-agent Assembly,* grant no. 120-1201948-1941. The work is also supported through international project *IGRAMO, Improving GRAsping Movements by predictions based on Observation,* conducted by KTH, Stockholm and University of Zagreb. The authors gratefully acknowledge the support.

## REFERENCES

Paredis, J.: Coevolutionary Computation, Artificial Life Journal, Vol. 2, nr. 4, MIT Press / Bradford boks (1996)

LaValle, S.M.: Planning Algorithms. Cambridge University Press, Cambridge (2006)

Canny, J.: The Complexity of Robot Motion Planning. MIT Press, Boston (2006)

Erderman, M., Lozano-Perez, T.: On multiple moving objects. In: Proceedings of the IEEE Conference on Robotics and Automation, San Francisco, California, USA, pp. 1419-1424 (1986)

Latombe, J. C.: Robot Motion Planning. Kluwer Academic Publishers, Boston (1991)

Rana, A. S., Zalzala, M.S.: An Evolutionary Algorithm for Collision Free Motion Planning of Multi-arm Robots. In: IEEE Genetic Algorithms in Engineering Systems: Innovations and Applications, (1995)

Rana, A. S., Zalzala, M.S.: An Evolutionary Planner for Near Time-Optimal Collision-Free Motion of Multi-Arm Robotic Manipulators. In: UKACC International Conference on CONTROL, (1996)

Davidor, Y.: Genetic Algorithm and Robotics; A Heuristic Strategy for Optimization, World Scientific, Singapore (1991)

Solteiro Pires, E.J., Tenreiro Machado, J.A., Moura Oliveira, P.B.: Robot Trajectory Planning Using Multi-objective Genetic Algorithm Optimization. In: Proceedings of the GECCO, LNCS, Springer, Heidelberg (2004)

Toyoda, Y., Yano, F.: Optimizing Movement of A Multi-Joint Robot Arm with Existence of Obstacles Using Multi-Purpose Genetic Algorithm. Ind. Eng. Man. Sys.. 3, 78--84 (2004)

Venegas Montes, H.A., Raymundo Marcial-Romero, J.: An Evolutionary Path Planner for Multiple Robot Arms. In: Evo Workshops, LNCS, Springer, Heidelberg (2009)

Sims, K.: Evolving 3D Morphology and Behavior by Competition. In:Artificial Life IV Proceedings, MIT Press (1994)

Michalewicz, Z.: Genetic Algorithms + Data Structures = Evolution Programs. Springer Verlag, Berlin (1999)

Bucci, A.: Emergent Geometric Organization and Informative Dimensions in Coevolutionary Algorithms. PhD thesis, Brandeis University, (2007)

Panait, L., Luke, S., Harrison, J.F: Archive-based Cooperative Coevolutionary Algorithms. In: Proceedings of the GECCO, (2006)